WHAT IS CLAIMED IS:

1. An embedded-DRAM (dynamic random access memory) processor comprising:

a DRAM array comprising a plurality of random access memory cells arranged in rows and columns;

a set of row address registers;

one or more sets of registers, each of said sets of registers capable of being loaded or stored in response to a single latch signal; and

an instruction set which includes:

(i) at least one command to perform arithmetic on said row address registers;

(ii) a command to precharge (activate) rows pointed to by said row address registers;

(iii) a command to deactivate rows pointed to by said row address registers;

(iv) a command to load a plurality of words of a row designated by said row address registers into designated sets of data registers; and

(v) a command to load selected columns of rows pointed to by said row address registers into designated sets of data registers, said selection based on bits in a mask.

2. The embedded-DRAM processor according to Claim 1, further comprising:

first and second sets of functional units, said first and second sets of functional units having respective first and second instruction sets and capable of accessing said first and second register sets;

a command to select one of said first and second sets of registers to be an architectural set of registers accessible to said first set of functional units;

a command to deselect the other of said first and second sets of registers so that it is no longer an architectural register set accessible to said first set of functional units;

a command to select one of said first and second sets of registers to be an architectural set of registers accessible to said second set of functional units; and

a command to deselect the other one of said first and second sets of registers so that it is no longer an architectural register set accessible to said second set of functional units.

3. The embedded-DRAM processor according to Claim 1, further comprising:

first and second sets of functional units, said first and second sets of functional units having respective first and second instruction sets and accessing said first and second register sets; and

a command which selects one of said first and second sets of registers to be an architectural set of registers accessible to said first set of functional units, and, at the same time, deselects said one of said first and second sets of registers to be an architectural set of registers accessible to said second set of functional units.

4. The embedded-DRAM processor according to Claim 1, further comprising:

first and second sets of functional units, said first and second sets of functional units having respective first and second instruction sets and accessing said first and second register sets; and

whereby said first and second instruction sets are subsets of said instruction set of said embedded-DRAM processor.

5. The embedded-DRAM processor according to Claim 4, whereby said second functional unit is a multi-issue functional unit and further comprises:

a dispatch unit;

a plurality of functional units which each execute a respective instruction stream as dispatched by said dispatch unit.

6. The embedded-DRAM processor according to Claim 1, further comprising a plurality of DRAM arrays.

7. The embedded-DRAM processor according to Claim 1, further comprising:

at least one functional unit;

whereby said one or more sets of registers comprise a plurality of register files, each said register file comprising a parallel access port operative to load or store contents of said register file in a single cycle from or to a DRAM row as selected by said row-address register, each said register file further comprising at least a second access port operative to transfer data between said functional unit and a selected subset register in said register file.

8. The embedded-DRAM processor according to Claim 7, further comprising:

a second functional unit;

whereby said first functional unit executes a first command to perform logical processing on the contents one or more registers within a selected active one of said register sets, and said second functional unit a executes second command to parallely transfer data between a selected inactive one of said register sets and said DRAM array.

9. The embedded-DRAM processor according to Claim 8, whereby said first and second functional units execute said first and second commands substantially contemporaneously.

10. The embedded-DRAM processor according to Claim 8, further comprising:

a first software module comprising a set of data manipulation commands, said first software module executed by said first functional unit; and

a second software module comprising a set of parallel data transfer commands, said second software module being executed by said second functional unit;

whereby said second software module operates in support of said first software module to prefetch data from said DRAM array into one of said register files in advance of said data being needed by said first software module.

11. The embedded-DRAM processor according to Claim 10, wherein:

said first software module contains an instruction that reference registers within an architectural register set visible to said first functional unit, whereby said architectural register set corresponds to at least partially to said one of said register files that is in an active state;

said second software module contains instructions that cause data to be transferred between an inactive register set and said DRAM array, and second software module also executes a command to toggle a selected register set between said active and inactive states.

12. The embedded-DRAM processor according to Claim 1, further comprising:

first and second sets of functional units, said first and second sets of functional units having respective first and second instruction subsets;

whereby said first instruction subset includes said command (i) and the second instruction subset includes said commands (ii), (iii), (iv) and (iv).

13. An embedded-DRAM (dynamic random access memory) processor comprising:

a DRAM array comprising a plurality of random access memory cells arranged in rows and columns;

a row address register;

one or more sets of data registers, each of said sets of data registers capable of being loaded or stored in response to a single latch signal;

a bit mask to select one or more data locations within at least one of said register sets; and

an instruction set which comprises at least:

(i) a command to perform arithmetic on said row address register;

(ii) a command to precharge (activate) a row pointed to by said row address register;

(iii) a command to deactivate a row pointed to by said row address register;

(iv) a command to load a set of selected elements of the row pointed to by said row address register into a selected set of said data registers, said selection based on bits in said bit mask.

14. The embedded-DRAM processor according to Claim 13, whereby said load command causes an entire row that was previously precharged to be loaded.

15. The embedded-DRAM processor according to Claim 13, whereby said load command causes a subset of a row that was previously precharged to be loaded.

16. An embedded-DRAM (dynamic random access memory) processor comprising:

a DRAM array comprising a plurality of random access memory cells arranged in rows and columns;

a row address register;

first and second registers files, each of said register files having a plurality of data registers capable of being loaded or stored in response to a single latch signal, each of said register files also being capable of being placed into an active state and an inactive state;

a bit mask to select one or more locations within at least one of said register files; and

an instruction set which comprises at least:

(i) a command to perform arithmetic on said row address register; and

(ii) a command to load a set of selected elements of the row pointed to by said row address register into a selected set of

said data registers, said selection based on bits in said bit mask.

17. The embedded-DRAM processor of Claim 16, whereby the instruction set further comprises:

(iii) a command to toggle a register set between said active and inactive states.

18. The embedded-DRAM processor of Claim 17, whereby said toggle command causes said first register file to toggle from the inactive state to the active state and also causes the second register file to toggle from the active state to the inactive state.

19. The embedded-DRAM processor of Claim 16, whereby the instruction set further comprises:

(iii) a command to manipulate the bits in the bit mask.

20. The embedded-DRAM processor according to Claim 16, further comprising:

first and second sets of functional units, said first and second sets of functional units having respective first and second instruction sets and capable of accessing said first and second register sets; and

said instruction set further comprises at least:

(iii) a command to select one of said first and second sets of registers to be an architectural set of registers accessible to said first set of functional units; and

(iv) a command to select one of said first and second sets of registers to be an architectural set of registers accessible to said second set of functional units.

21. The embedded-DRAM processor according to Claim 20, said instruction set further comprising:

(v) a command to deselect the other of said first and second sets of registers so that it is no longer an architectural register set accessible to said first set of functional units; and

(vi) a command to deselect the other one of said first and second sets of registers so that it is no longer an architectural register set accessible to said second set of functional units.

22. The embedded-DRAM processor according to Claim 20, whereby at least one of said sets of functional units contains a single functional unit.

23. An embedded-DRAM (dynamic random access memory) processor comprising:

a DRAM array comprising a plurality of random access memory cells arranged in rows and columns;

a row address register;

first and second registers files, each of said register files capable of being loaded or stored in response to a single latch signal, each of said register files also being capable of being placed into an active state and an inactive state; and

first and second of functional units, said first and second functional units having respective first and second instruction sets and capable of accessing said first and second register sets;

whereby said first and second registers files comprise a parallel access port operative to parallely transfer contents of said register file between a DRAM row as selected by said row-address register, each said register file further comprising at least a second access port operative to transfer data between a selected register file and said second functional unit;

whereby said first instruction set comprises at least:

(i) a command to manipulate data in a data register within a register file; and

whereby said second instruction set comprises at least:

(ii) a command to perform arithmetic on said row address register;

(iii) a command to load the row pointed to by said row address register into a selected set of registers of said register files.

24. The embedded-DRAM processor according to Claim 23, whereby said first and second functional units each respectively execute a command from said first and second instruction sets substantially contemporaneously.

25. The embedded-DRAM processor according to Claim 24, further comprising:

a first software module comprising a data manipulation commands drawn from said first instruction set, said first software module executed by said first functional unit; and

a second software module comprising a parallel data transfer command drawn from said second instruction set, said second software module being executed by said second functional unit;

whereby said second software module operates in support of said first software module to prefetch data from said DRAM array into one of said register files in advance of said data being needed by said first software module.

26. The embedded-DRAM processor of Claim 23, whereby the second instruction set further comprises:

(iv) a command to toggle a register set between said active and inactive states.

27. The embedded-DRAM processor of Claim 26, whereby said toggle command causes said first register file to toggle from the inactive state to the active state and also causes the second register file to toggle from the active state to the inactive state.

28. An embedded-DRAM (dynamic random access memory) processor comprising:

a DRAM array comprising a plurality of random access memory cells arranged in rows and columns;

first and second dual-port registers files, each of said register files capable of parallely transferring data between a row of said DRAM array, each said register files also being capable of being placed into an active state and an inactive state; and

first and second of functional units, said first and second functional units having respective first and second instruction sets;

whereby said first instruction set comprises at least:

(i) a command to manipulate data in a data register within a register file; and

whereby said second instruction set comprises at least:

(ii) a command to unidirectionally transfer data between a row of said DRAM array and a selected inactive data register file;

(iii) a command to place said inactive register file into said active state, whereby when the register set is activated, it becomes an architectural register set of said first functional unit.

29. The embedded-DRAM processor of Claim 28, whereby said command to unidirectionally transfer data causes data to be transferred from a row of the DRAM array to said selected inactive data register file.

30. The embedded-DRAM processor of Claim 28, whereby said command to unidirectionally transfer data causes data to be transferred from said selected inactive data register file to a row of the DRAM array.

31. The embedded-DRAM processor of Claim 28, whereby the said command to place the selected inactive register file into the active state is a command that also causes the remaining register file to toggle from the active state into the inactive state.

32. The embedded-DRAM processor of Claim 28, further comprising:

at least one additional register file;

whereby the said command to place the selected inactive register file into the active state is a command that also causes a selected other register file to toggle from the active state into the inactive state.

33. The embedded-DRAM processor of Claim 28, further comprising:

at least one a row address pointer, whereby at least one command in said second instruction set uses said row address pointer to identify said selected register file; and

the second instruction set further comprises:

(iv) a command to manipulate the at least one row address pointer.

34. The embedded-DRAM processor of Claim 28, further comprising:

at least one a bit mask; and

the second instruction set further comprises:

(iv) a command to move a subset of elements between a selected register file and a selected row of said DRAM array, whereby said subset is identified by said bit mask.

35. The embedded-DRAM processor according to Claim 28, whereby said first functional unit is a multi-issue functional unit and further comprises:

a dispatch unit;

a plurality of functional units that each execute a respective instruction stream as dispatched by said dispatch unit.

36. The embedded-DRAM processor according to Claim 28, further comprising:

a first software module comprising a set of data manipulation commands drawn from said first instruction set, said first software module executed by said first functional unit; and

a second software module comprising a set of parallel data transfer commands drawn from said second instruction set, said second software module being executed by said second functional unit;

whereby said second software module operates in support of said first software module to prefetch data from said DRAM array into one of said register files in advance of said data being needed by said first software module.

37. The embedded-DRAM processor according to Claim 28, wherein:

said first software module contains an instruction that reference registers within an architectural register set visible to said first functional unit, whereby said architectural register set corresponds to at least partially to said one of said register files that is in an active state;

said second software module contains instructions that cause data to be transferred between an inactive register set and said DRAM array, and second software module also executes a command to toggle a selected register set between said active and inactive states.

38. The embedded-DRAM processor according to Claim 28, whereby each of said register files contain a number of words, N, matched to the number of words in of a row of said DRAM array, and said parallel load and store operations involve moving said selected row in its entirety to said selected register file.

39. The embedded-DRAM processor according to Claim 28, further comprising:

a mask and switch unit interposed between said DRAM array and at least one of said register files.

40. The embedded-DRAM processor according to Claim 28, whereby said second set of instructions comprises:

a command to cause data to be moved from one register to another within a given one of said register files.

41. The embedded-DRAM processor according to Claim 28, whereby said second instruction set is used to implement an intelligent caching scheme, whereby said register files act as a cache and said second set of instructions are executed in lieu of a standard cache that maintains most recently used data and enforces a set associative or a direct-mapped caching policy.

42. The embedded-DRAM processor according to Claim 28, further comprising:

an instruction register coupled to receive instructions from said instruction set, said instruction register operative to hold an instruction to be executed by said data assembly unit; and

a local program memory coupled to said instruction register;

whereby said second functional unit corresponds to a data assembly unit, and said data assembly unit receives an instruction from said second instruction set that causes a separate control thread of instructions to be accessed from said local program memory and executed by said data assembly unit.

43. The embedded-DRAM processor of claim 42, further comprising:

a prefetch unit that prefetches instructions from the first and second instruction sets from a single very long instruction word (VLIW) instruction memory; and

a dispatch unit that dispatches instructions from the first instruction set to the functional units and dispatches instructions from the second instruction stream to the data assembly unit.

44. The embedded-DRAM processor according to Claim 28, whereby said second functional unit monitors execution activity of instructions in said first instruction set and said second instruction set further comprises:

(iv) a command to precharge a row of the DRAM array;

whereby the second functional unit executes a speculative precharging to prevent program delays due to DRAM row precharging.

45. An embedded-DRAM (dynamic random access memory) processor comprising:

a DRAM array comprising a plurality of random access memory cells;

first and second dual-port registers files, whereby the first port of each of said register files is a parallel access port and is parallely coupled to said DRAM

array, each said register file is capable of being placed into an active state and an inactive state;

at least one functional unit that executes a first program, said functional unit coupled to said second port of said register files, said functional unit responsive to commands exclusively involving architectural register operands that map onto to the registers within a register file that is in the active state;

a data assembly responsive to an instruction set comprising at least:

(i) a command that causes data to be moved between the DRAM array and a register file that is in the inactive state; and

(ii) a command that causes said register file in the inactive state to assume the active state and said register file in the active state to assume the inactive state.

46. In a digital processor comprising a DRAM array having a plurality of random access memory cells arranged in rows and columns, a set of row address registers, one or more sets of registers each capable of being loaded or stored in response to a latch signal, a method of processing data comprising:

performing arithmetic on said row address registers;

precharging (activating) rows pointed to by said row address registers; and

loading a plurality of words of a row designated by said row address registers into designated sets of data registers.

47. The method of Claim 46, further comprising deactivating rows pointed to by said row address registers.

48. In a digital processor comprising a DRAM array having a plurality of random access memory cells arranged in rows and columns, a set of row address registers, one or more sets of data registers each capable of being loaded or stored in response to a latch signal, a method of processing data comprising:

performing arithmetic on said row address registers;

precharging (activating) rows pointed to by said row address registers; and

loading selected columns of rows pointed to by said row address registers into designated sets of said data registers, said selection based on bits in a mask.

49. In a digital processor comprising a DRAM array having a plurality of random access memory cells arranged in rows and columns, first and second dual-port registers files each capable of (i) parallel transfer of data between a row of said DRAM array, and (ii) being placed into an active state and an inactive state, and first and second functional units, a method for processing data comprising:

manipulating data in a data register within a register file using said first functional unit; and

using said second functional unit;

(a) unidirectionally transferring data between a row of said DRAM array and a selected inactive data register file; and

(b) placing said inactive register file into said active state, whereby when the register file is activated, it becomes an architectural register set of said first functional unit.